An AOC WS LSI Concept Paper

Complex Systems and Evolutionary Engineering

Y. Bar-Yam
New England Complex Systems Institute
24 Mt. Auburn St., Cambridge, MA 02459

M. L. Kuras
The MITRE Corporation
Bedford, MA 01730

September, 2003

This document is for informational purposes only. It is not directing nor dictating potential solutions to the AOC WS LSI effort. Rather its purpose is to provide additional information in support of potential future Government interaction with industry.

Complex Systems and Evolutionary Engineering

Y. Bar-Yam
New England Complex Systems Institute
24 Mt. Auburn St., Cambridge, MA 02459

M. L. Kuras
The MITRE Corporation
Bedford, MA 01730

1.0 Introduction

AOCs (Air and Space Operations Centers) are complex systems.

This does not simply mean that they are hard to grasp in their entirety (individually or collectively). It means some very specific, tangible things. It means, for example, that AOCs continually evolve (often in unexpected but acceptable ways) and that no two of them are exactly alike. This, and additional characteristics, separate AOCs as systems from other, more familiar, systems. Additional distinguishing characteristics are taken up below.

For the upcoming acquisition, however, what is even more important is the process by which the Government now expects that these AOCs will be provisioned, integrated, standardized, modernized and operated. This is because the Government-selected LSI (the Lead System Integrator) will be an embedded part of that process, and that process is different in many respects from the one presently associated with the acquisition or development of familiar systems.

2.0 A Metaphor

A metaphor is frequently used to introduce the differences involved. The metaphor contrasts the behavior of a "watchmaker" with that of a "gardener." The Government-selected AOC LSI will behave more like the gardener.

A watchmaker succeeds by meticulous attention to every detail. Precision and accuracy are nearly synonymous. If even one gear is slightly askew, the entire timepiece is at risk. A gardener, on the other hand, focuses attention on the whole "garden." Individual plants are selected and then "nurtured" by the gardener, but are otherwise left to attend to the details of their individual growth. A gardener monitors and provides essentials such as

water and fertilizer, but then the individual plants employ such things as appropriate to their individual circumstances. The gardener constantly "weeds," identifying and removing plants behaving at variance with the overall well being of the garden. The gardener will recognize (and even stimulate) new variations of plants and continually rebalance the overall arrangement of the garden to accommodate the most attractive and fruitful.

What the LSI will do, to ensure the realization of the AOCs as complex systems, is not *ad hoc* or impromptu, just as a good gardener does not behave in such a fashion if the result is going to be a flourishing garden.

3.0 A Vignette

In order to get a more realistic sense of the differences involved, a brief and partial vignette is presented. The timeframe is the summer of 2006. The situation and the events are hypothetical, but they are notionally representative nonetheless.

One of the AOCs is in Qatar. It has been operational since 2003. It has been operational around the clock (24 x 7) since then without interruption, but it is a very different AOC today (in the summer of 2006) than it was in the spring of 2003. Some of the equipment first installed in 2003 is still there and still being used, but much of the equipment – and more importantly perhaps, the software – is different. And so is the way much of that equipment (old or new) is being used. Few of the people charged with its operational functioning in the summer of 2006 have been on-station for more than two months. The primary operational mission of this AOC is the preemption of terrorist acts anywhere in its theater of operations (which includes both Afghanistan and Iraq). Preemption involves reaction *before* unfolding terrorist acts can actually occur. Meanwhile, many "routine" activities must still be attended to (the scheduling of airlift, the disposition of air patrols, etc.) The integrated capability to do all of these things concurrently (as well as the training to exploit this capability) was not, however, fully appreciated when put to use for the first time (and probably still isn't). More importantly, this capability and its associated training were only vaguely understood in the spring of 2004 when the AOC LSI was selected. Nonetheless, the LSI was able to successfully package, qualify and install all of the changes needed (down to the last IP address and fiber optic cable splice).

But this is not the only operational AOC. Another is located in Korea. Here the operational focus is different. Attention is focused on the strict enforcement of a recent multinational treaty and the expeditious delivery of humanitarian relief to remote and normally inaccessible regions. Many of the "routine" activities are the same as in the Qatar AOC, but the balance of activities and capabilities is different. Some are even unique to this one Korean AOC. The operational staff has been on-station longer, but even here many of them are newcomers. Some have served in other AOCs. They expect (and get) a familiar environment that leverages their prior training and experience.

Yet another AOC did not exist (and wasn't even envisioned) until three months ago. It is ship borne and operating in the South Atlantic, near Antarctica. Nonetheless, the LSI was able to successfully package, qualify and install this newest of AOCs starting from a "partial" AOC that had been in use as a CONUS training facility in Nevada (and which the LSI is now replacing).

Much of the hardware and software and training that went into the realization of these three AOCs (and of the other AOCs as well) was familiar to the LSI in 2004. Some of it the LSI may have in fact created in the first place. But much of it came from other commercial sources and from other Government-run programs. In the interval since 2004, much of this materiel has been "debugged" or "technologically refreshed." The LSI has acted to ensure that these changes have been packaged, qualified, and installed in all of the AOCs without disrupting their mission effectiveness or readiness.

But other, more dramatic, changes have become available as well, and are constantly being incorporated into some or all of the AOCs. These changes do not simply represent better or more current implementations of existing functionality and capacity. They involve new or enhanced expressions of functionality that meld "seamlessly" and "effortlessly" with already existing ones. (The existing functionality does not have to change or be changed to accommodate the new functionality.) The key to this blending of the new with the current is the existence of a common "infostructure" in all of the AOCs.[1] An important aspect of the qualification of any change to the AOCs is the LSI's certification that changes behave in accordance with the "standards" implied by this common "infostructure."

But this "infostructure" itself must be capable of gradual improvement. And even more importantly, qualification for incorporation into any AOC requires more than just this minimal certification. The changes must actually interwork "seamlessly and effortlessly" with other elements of functionality in the AOCs in order to produce enhanced capability. Simply drawing current without blowing fuses isn't enough.

In order to produce a steady stream of such improvements to the capabilities of the AOCs, the LSI also oversees a "developmental environment." This environment serves two purposes. It does so by emulating (not just simulating) actual (or at least well-approximated) operational and systemic conditions in the AOCs. The LSI employs this environment first to prepare, package, and pre-qualify all changes prior to their incorporation into one or more of the AOCs. Second, the LSI maintains this environment for the benefit of other commercial firms (as well as other Government agencies) so that these "third parties" can explore the utility of prospective changes, and then pursue and resolve the many practical issues associated with the realization of their prospective changes – all with little or no direct participation by the LSI.

[1] This "infostructure" is not strictly hypothetical. It has already begun to take shape. It is the subject of a companion concept paper.

The viability of many of these prospective changes, as well as their practical realization in terms of hardware and software development, will frequently depend on the ongoing interworking of more than one developer, long before the fruits of such efforts actually appear in AOCs. The "developmental environment" functions to support such interactions among developers (and not just among their products). Such interactions cannot be scripted in advance but they can be policed, and this is something that the LSI has been doing on behalf of the Government.

One particular change (in the summer of 2006) is getting a lot of attention. It is the functionality that finally permits the effective preemption of most terrorist acts. It is functionality that utterly depends on functionality introduced earlier to successfully prosecute "time sensitive targets." (Neither change was understood in any practical depth in the spring of 2004 when the AOC LSI was selected.)

The heart of this new functionality is a set of software applications developed by a small start-up that didn't even exist until early in 2005. These applications, in turn, embody a number of data mining, pattern recognition, and "stock management" algorithms developed (more or less independently) by DARPA, the DIA, and several commercial firms over the previous ten years. This new firm, Normatics Inc., is going to reap the financial rewards for their effort and eventual success (and that will reward in turn their investors).

Normatics, Inc. was not alone in 2005 when they recognized the opportunity they have now successfully seized. Several other commercial firms (and a Government agency) also did. They too formulated solutions and then partially or wholly implemented their respective prospective AOC changes in the context of the LSI's developmental environment. For a number of reasons, all but the Normatics approach fell by the wayside…

The Normatics Inc. change is remarkable in another respect. As a consequence of introducing this change, there are now a number of additional changes happening to the "workflow" in the Qatar AOC. Substantial fractions of the "intel" and "time critical targeting" cells are now being reorganized into a "preemption" cell; this AOC has begun to publish (on a six hourly basis) a remarkably detailed "predictive battlespace picture" for use by the "current ops" and "planning" cells; etc.

Needless to say, the AOC LSI could not have been made aware of these future changes when it was selected in 2004. Nonetheless, the LSI not only successfully introduced the Normatics Inc. innovation into the Qatar AOC, they are even now working with the operational staff of that AOC to facilitate the new workflow changes in terms of system applications, loadings, etc.; and they have already begun to investigate (in their developmental environment) how these workflow changes can be mimicked in some or all of the other AOCs.

4.0 Complex System Evolutionary Engineering

But more than metaphors and vignettes are available to convey the many aspects of complex systems and their development. Complex System Evolutionary Engineering (CSEE) embodies this knowledge. Simply stated, CSEE is the deliberate and accelerated mimicry of the processes that drive "natural evolution." [1, 2]

When the complexity of a system, captured and framed through a multiscale analysis [3-5] of its interdependences, exceeds certain bounds, it is necessary to adopt an approach that mimics and accelerates natural evolution. Various contemporary approaches have already incorporated aspects of evolutionary processes. This includes incremental change [6], experience based learning [7, 8], spiral development and evolutionary acquisition [9], and adaptive or extreme programming [10]. However, a CSEE perspective provides a larger conceptual framework in which to understand how repetitive incremental change can safely produce both rapid innovation and increased overall complexity [1, 2]. This total evolutionary framework is most commonly associated with the formation and behavior of biological organisms, but it is also evident in the development of free market economies, ecosystems, social organizations, etc. [11]

Understanding a complex system approach to design and implementation involves recognizing the many differences between the natural evolutionary process and traditional engineering practices. CSEE employs, among others, the following key concepts, that may be contrasted to traditional engineering practices:

- Focus on creating an environment and process rather than a product.
- Continually build on what already exists.
- Operational components are modifiable in situ.
- Operational systems include multiple versions of functional components.
- Utilize multiple parallel development processes.
- Evaluate experimentally in-situ.
- Increase utilization of more effective components, gradually.
- Effective solutions to specific problems cannot be anticipated.
- Traditional system engineering should be used for not-too-complex components.

4.1 Focus on creating an environment and process rather than a product:

Ongoing change in a system is the underlying mechanism of creation, not the formulation and execution of plans. Encouraging and safeguarding this ongoing change and monitoring its outcomes are the absolute essentials of an evolutionary-based process.

4.2 Continually build on what already exists:

Off-line engineering of complex systems is impractical because the complexities of their environment and true functional requirements do not permit practical specification or

testing prior to implementation. In complex systems, correct expectations and testing both depend on the immediate consequences of current operations.

4.3 Individual components must be modifiable in situ

The interdependencies between system components must be such that individual components can be modified in situ. In practice this requires the following point.

4.4 Operational systems include multiple versions of functional components

Complex systems should be understood as populations rather than as rigid assemblies of unique components. Individual components can overlap substantially in terms of both functionality and interaction. Evolutionary processes impact both populations and individuals. Redundancies are not always unwanted inefficiencies.

4.5 Utilize multiple parallel development processes

The existence of populations of components allows multiple parallel efforts to explore modifications that might (but that are not guaranteed) to improve system components and/or total system capability.

4.6 Evaluate experimentally in-situ

Testing and experimentation increasingly overlap. Off-line qualification testing becomes a prelude to active field testing for components in a large variety of operational environments. Results (including unexpected results) are ratified or rejected as they occur based on then-current overall system capability.

4.7 Increase utilization of more effective components, gradually

The replacement of components cannot be abrupt as testing is never complete and operation is continuous. Augmentation and parallel operation is the preferred approach.

4.8 Effective solutions to specific problems cannot be anticipated

Specification efforts cannot assume that the most efficient or even effective solutions can be anticipated in advance of an exploration and discovery process involving multiple parallel development efforts. As a result, complex solutions cannot be successfully specified in advance. This is increasingly apparent the more complex a solution must be

in its totality in order for it to succeed even marginally. Moreover, this remains the case no matter how long a problem is worked and progressively better solutions are found.

5.0 The "Integration" of Complex Systems

In order to operate a CSEE process, the concept of integration must be radically rethought. A systematic and effective application of the ideas in this paper involves a "paradigm shift" from "complete system specification" to the creation of environments that are conducive to ongoing change in components of systems while supporting the more or less constant evaluation of their overall effectiveness through virtual as well as real world testing.

An LSI is not expected to provide specifications of components or interfaces. Instead, the LSI establishes and oversees an environment in which components are gradually but continually conceived, implemented, fielded, and evaluated. Specifics of the environment and the relationships among the LSI, the component builders and their respective engineering platforms can only be discussed in terms of evolutionary ideas rather than specifications and implementations. The "total" security that is possible from a "complete" specification is not possible. Moreover, higher risk accompanies the greater gain that comes from the partial absence of constraints imposed by early specification. This risk is balanced by heightened care in the operation of the developmental and operational environments that increasingly overlap. Hence, the role of the LSI, while different, remains crucial.

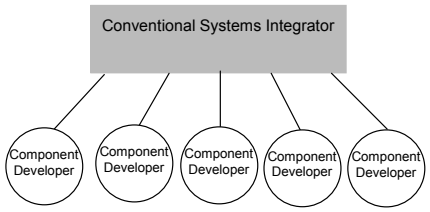

Figure 1

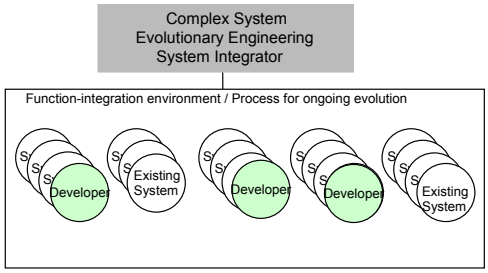

Figure 2

5.1 The Developmental Environment

One facet of the difference between conventional systems integration and the complex system approach is shown in Figures 1 and 2. The conventional approach (Figure 1) places the LSI in direct hierarchical interaction with each of the component developers. The LSI is responsible for defining or ratifying all of the interfaces among the individual components of the system generated by the developers, and for (at least) the final stages of assembling the components into a system. In the complex system approach (Figure 2), the LSI oversees an environment in which the components of the overall system can be operated as much as possible as though a part of the "real" system, and in which the individual component developers interact with one another directly in order to define and realize the interfaces among their components.

The developmental environment provides:

- A process of evaluation
- A mechanism of reward
- Safety constraints
- Rules for cooperation and competition

These rules, constraints, mechanisms and processes of the environment are themselves dynamic.

5.1.1 A process of evaluation

Evaluation involves the determination of the relative contributions of components to the overall capability of a system as those contributions actually occur. In more traditional situations, very similar evaluations are attached to outcomes that *might* occur in the future.

5.1.2 A mechanism of reward

The responsible components and their developers receive appropriate rewards to both compensate for *prior* efforts and to encourage further development efforts.

5.1.3 Safety constraints

The dynamic of component augmentation and/or replacement, as well as normal operation, are subject to adequate safeguards that ensure system robustness and safety in both operational and developmental contexts.

5.1.4 Rules for cooperation and competition

Participants in the developmental environment are subject to contractually binding rules that govern both cooperation and competition. These are expressed as developmental precepts of the developmental environment, and serve to stimulate mutual discovery and interaction among the participants without specifying detailed interactions among them or their creations. The processes of component incorporation and evaluation are also subject to rules that ensure the integrity of the environment.

5.2 Applicability

CSEE does not replace traditional system engineering. CSEE is relevant to the level of system development at which complexity can no longer be managed exclusively through anticipation and attention to detail.

Traditional system engineering continues to be relevant and valuable for levels of system complexity when both anticipation and attention to detail suffice (as will continue to be the case for many of the components of a complex system). The overall development of the AOCs is recognized by the Government to be at a level of complexity exceeding such a threshold.

6.0 Roles

The upcoming selection of an AOC LSI will contractually fix the "role" of the LSI in the provisioning, integration, standardization, and modernization of the AOCs. The AOC LSI selection will do this rather than attempt to fix the "outcomes" that an LSI might be expected to deliver at some future date.

What might that LSI "role" be? (And what roles won't be assumed by the LSI?) How should this LSI-assumed role be contractually fixed?

During this early phase of the selection process, answers to these questions will be developed in consultation with the commercial community. A complete set of role players in an AOC complex system development might be as follows.

- Operators: the people who use the equipment in an AOC to perform military tasks;
- Trainers: the people who instruct operators in the conduct of their military tasks, and in the use of the AOC's equipment;
- Installers and Maintainers: the people who install and maintain AOC equipment in operational settings;
- Administrators: the people who maintain the operational configuration of all equipment in the AOCs;
- Component developers: the people who conceive of and then mature new or modified pieces of equipment (and software) intended or destined for AOCs;
- Workflow developers: the people who conceive of and then mature operational doctrine, TTPs, etc.;
- Appraisers: the people who characterize without judgment the continuous operation of the AOCs (both people and equipment);
- Referees: the people who monitor activities in the AOCs' developmental and operational environment and intervene to maintain its integrity;

- Judges: the people who evaluate the characterizations of actual AOC operation and assign credit for improvements in terms of individual component and workflow developers.

An LSI might well be asked to provide all of the Installers and Maintainers, Administrators, Appraisers and Regulators for the AOCs as complex systems. The precise contractual expression of such an LSI role will reflect the insights to be gained from a thorough appreciation of the discussion in sections 4 and 5.

7.0 Complex and not-so-complex systems

Given the above, it is now possible to examine the distinction between highly complex systems and the more familiar sort. Highly complex systems go by many names today. So do the more traditional or familiar systems. The term "enterprise" is frequently associated with highly complex systems, and the term "product" is usually understood to refer to a familiar type of system.

An enterprise and a product are both systems. They are both examples of the arrangement of many separate parts into one whole. Enterprises and products are also different in many respects, however, and that is why the Government now expects that they will be developed differently – but not wholly differently. Familiar and new processes in new proportions will be applied to the realizations of the Government's enterprises.

Products are reproducible – exactly reproducible. There are almost always many instances of the "same" product. Enterprises are not, however; no two are ever exactly the same.

Products are built to pre-conceived specifications. Enterprises constantly change and cannot be fully pre-specified. For products, extreme care is given first to getting the specifications right and then translating those specifications into realizations. Traditional system engineering has emerged to give rigor to this process. In the case of the enterprise, care is given to getting the conditions right so that an enterprise can emerge on its own, flourish, and evolve. This is where CSEE is focused.

Products have well-defined boundaries. This is both necessary and inevitable. Such boundaries aid in setting accountability, for example. Enterprises, on the other hand, have ill-defined, ambiguous, and shifting boundaries. This enables enterprises, for example, to adjust as their circumstances change.

In the development of products, people work to continually remove unwanted possibilities. This is a major focus of traditional testing. Enterprises, on the other hand, continually seek or even create new possibilities in order to exploit them to advantage. This entails a reordering of priorities in the blending of experimentation and testing.

External agents are required to integrate products from specified components. Enterprises, on the other hand, constantly integrate and re-integrate themselves, drawing on both existing and new components, while gradually discontinuing reliance on still others.

The development of products has a well-defined and attainable end. Everything works as expected at this point. Enterprise development never ends. Enterprises constantly change. Their development is an integral facet of their operation. To stop their development would be to stop their operation.

The development of products strives to eliminate all sources of internal conflict. Total harmony and cooperation, complete synchronization, and so on, are the ultimate goals and efficiency is the metric. The successful development of enterprises, on the other hand, absolutely depends on both cooperation *and* competition (friction, disharmony, striving for different objectives, striving to control or use the same resources, etc.). The complete elimination of competition causes enterprises to collapse in a spiral of increasing efficiency.

There is always a single and ultimate authority for all decision making in product development. This authority is almost always carefully and hierarchically distributed for reasons such as timeliness and efficiency. In enterprises, there is no such ultimate authority. Decision making is always localized and circumscribed to some extent, and there is always a tradeoff between the span of influence and control, and the depth of that influence and control.

8.0 Concluding remarks

The introductory metaphor and vignette were not meant to be prescriptive; they are entirely illustrative. There is more than one way in which the principles of CSEE can be applied to the acquisition and development of the AOCs. The eventual success of the AOC LSI will reflect this, as the LSI applies its unique understanding of CSEE to the specifics of the actual AOCs. One element of the pending selection is to provide the government with an early appreciation of these eventualities.

There is a growing body of information available on this topic of complexity and its relevance to system analysis and synthesis.. A short concept paper such as this can do little more than draw a reader's attention to the subject. A fuller appreciation can be developed by tapping resources such as the following.

8.1 Links to Resources on Military Complex Systems:

Clausewitz and Complexity
CCRP, Command and Control Research Program
ISAAC - Irreducible, Semi-Autonomous Adaptive Combat

8.2 Links to Resources on Complex systems:

New England Complex Systems Institute
Santa Fe Institute

References:

[1] Y. Bar-Yam, Enlightened Evolutionary Engineering / Implementation of Innovation in FORCEnet, Report to Chief of Naval Operations Strategic Studies Group, 2002 (Brief 2000).

[2] Y. Bar-Yam, "When Systems Engineering Fails – Toward Complex Systems Engineering," IEEE (in press).

[3] Y. Bar-Yam, "Unifying Principles in Complex Systems" in Converging Technology (NBIC) for Improving Human Performance, M. C. Roco and W. S. Bainbridge, Dds., Kluwer, 2003.

[4] Y. Bar-Yam, "General Features of Complex Systems" in UNESCO Encyclopedia of Life Support Systems (in press).

[5] Y. Bar-Yam, Dynamics of Complex Systems, Perseus, Reading, MA, 1997.

[6] Standish Group International, The CHAOS Report, 1994.

[7] G. S. Lynn, J. G. Morone and A. S. Paulson, "Marketing and discontinuous innovation: The probe-and-learn process," California Management Rev., Vol. 38, No. 3, pp. 8–36, 1996.

[8] R. W. Veryzer, "Discontinuous innovation and the new product development process," J. Product Innovation Management, Vol. 15, pp. 304–321, 1998

[9] DoD Directive 5000.1, "The Defense Acquisition System," May 12, 2003.

[10] See e.g. Agile Software Development, CrossTalk, Vol. 15, No. 10, Oct. 2002.

[11] J. C. Herz, "The Allure of Chaos," The Industry Standard, June 25, 2001.